2 Accessing and analysing corpus data

2.1 Introduction

The role of corpus data in linguistics has waxed and waned over time. Prior to the mid-twentieth century, data in linguistics was a mix of observed data and invented examples. There are some examples of linguists relying almost exclusively on observed language data in this period. Studies in field linguistics in the North American tradition (e.g. Boas 1940) often proceeded on the basis of analysing bodies of observed and duly recorded language data. Similarly, studies of child language acquisition often proceeded on the basis of the detailed observation and analysis of the utterances of individual children (e.g. Stern and Stern 1907) or else were based on large-scale studies of the observed utterances of many children (Templin 1957). From the mid-twentieth century, the impact of Chomsky's views on data in linguistics promoted introspection as the main source of data in linguistics at the expense of observed data. Chomsky (interviewed by Andor 2004: 97) clearly disfavours the type of observed evidence that corpora consist of:

> Corpus linguistics doesn't mean anything. It's like saying suppose a physicist decides, suppose physics and chemistry decide that instead of relying on experiments, what they're going to do is take videotapes of things happening in the world and they'll collect huge videotapes of everything that's happening and from that maybe they'll come up with some generalizations or insights. Well, you know, sciences don't do this. But maybe they're wrong. Maybe the sciences should just collect lots and lots of data and try to develop the results from them. Well if someone wants to try that, fine. They're not going to get much support in the chemistry or physics or biology department. But if they feel like trying it, well, it's a free country, try that. We'll judge it by the results that come out.

The impact of Chomsky's ideas was a matter of degree rather than absolute. Linguists did not abandon observed data entirely – indeed, even linguists working broadly in a Chomskyan tradition would at times use what might reasonably be described as small corpora to support their claims. For example, in the period from 1980 to 1999, most of the major linguistics journals carried articles which were to all intents and purposes corpus-based, though often not self-consciously

so. *Language* carried nineteen[1] such articles, *The Journal of Linguistics* seven,[2] and *Linguistic Inquiry* four.[3] But even so there is little doubt that introspection became the dominant, indeed for some the only permissible, source of data in linguistics in the latter half of the twentieth century. However, after 1980, the use of corpus data in linguistics was substantially rehabilitated, to the degree that in the twenty-first century, using corpus data is no longer viewed as unorthodox and inadmissible. For an increasing number of linguists, corpus data plays a central role in their research. This is precisely because they have done what Chomsky suggested – they have not judged corpus linguistics on the basis of an abstract philosophical argument but rather have relied on the results the corpus has produced. Corpora have been shown to be highly useful in a range of areas of linguistics, providing insights in areas as diverse as contrastive linguistics (Johansson 2007), discourse analysis (Aijmer and Stenström 2004; Baker 2006), language learning (Chuang and Nesi 2006; Aijmer 2009), semantics (Ensslin and Johnson 2006), sociolinguistics (Gabrielatos *et al*. 2010) and theoretical linguistics (Wong 2006; Xiao and McEnery 2004b). As a source of data for language description, they have been of significant help to lexicographers (Hanks 2009) and grammarians (see sections 4.2, 4.3, 4.6, 4.7). This list is, of course, illustrative – it is now, in fact, difficult to find an area of linguistics where a corpus approach has *not* been taken fruitfully.

The ubiquitous use of corpus data is actually not surprising, given a correct understanding of the analogy between linguistics and the physical sciences. Chomsky's view is at best somewhat naïve and at worst deliberately misleading. Contrary to his assertion, there are entire fields of natural science based not on laboratory experiments but on the collection, and subsequent analysis, of large amounts of observational data: astronomy, geology and palaeontology, to name but three. Even theoretical physics is reliant on real-world observation to confirm or deny its proposals. To give one famous example, it was precise observations of the orbit of the planet Mercury, and of the deflection of starlight by the gravity of the sun, that confirmed the accuracy of Einstein's general theory of relativity, one of the cornerstones of modern physics.[4]

So, unless we are willing to discard as invalid a large part of modern science, Chomsky's argument against corpus linguistics collapses. Just as observation of the universe through astronomy can help to prove the hypotheses of physicists such as Einstein, so observation of language through corpora can help linguists to understand language. But we must not confuse corpus data with language itself. Corpora allow us to *observe* language, but they are not language itself. Furthermore, we do not claim that corpora are the only tool that linguists should use to explore language – introspection and other data collection methods do have their role to play in linguistics. Indeed, without some ability to introspect, it is doubtful whether a linguist could ever formulate a question to ask of a corpus. Nonetheless, there is little doubt that, for the majority of researchers in linguistics, corpora are an indispensable source of evidence, and the tools that extract data from corpora are a substantial and transformative source of support. The utility of

corpus data, and of using it alongside other data, is well summarised by Fillmore (1992: 35):

> I have two observations to make. The first is that I don't think there can be any corpora, however large, that contain all of the areas of English lexicon and grammar that I want to explore; all that I have seen are inadequate. The second observation is that every corpus that I've had the chance to examine, however small, has taught me facts that I couldn't imagine finding out in any other way . . .

Fillmore's sensible methodological pluralism is now widespread, including among linguists who might previously have espoused a much more straightforwardly hostile position towards corpus linguistics. For example, Wasow (2002: 163), a theoretical linguist, observes:

> While data from corpora and other naturalistic sources are different in kind from the results of controlled experiments (including introspective judgment data), they can be extremely useful. It is true that they may contain performance errors, but there is no direct access to competence; hence, any source of data for theoretical linguistics may contain performance errors. And given the abundance of usage data at hand, plus the increasingly sophisticated search tools available, there is no good excuse for failing to test theoretical work against corpora.

2.2 Are corpora the answer to all research questions in linguistics?

If we consider the range of research questions that a corpus on its own allows us to address, we can imagine it as covering a subset of all the research questions that a linguist might ask. That subset will overlap with the subset of questions that a linguist can ask *without* a corpus, but it is almost certainly greater in size than that set. This is because *with* corpus data, we can take new approaches to a number of areas, including grammatical description (see Chapter 4) and even linguistic theory (see Chapters 6 and 7). Moreover, the set of questions that may be readily addressed using a corpus grows significantly as suitable tools for interrogating that data become available. It grows larger still if the corpus data we have at our disposal is suitably annotated with linguistic analyses, in such a way that linguistically motivated queries can be undertaken rapidly and accurately. For example, without a corpus, we could certainly examine, and seek to describe, the use of non-finite verbs in English – many scholars have, for instance O'Dwyer (2006: 58–9). However, the number of examples of non-finite verbs that we could base our investigation on would remain relatively small, being limited by the hand-and-eye techniques that we would need to find them. With a suitable corpus of English, we would have at

our disposal many thousands of words with which to conduct our study. Still, in the absence of tools to search the data rapidly, the exploitation of the data would be slow and prone to error – although we would probably be able to study a greater number of examples somewhat more effectively than we would without the corpus. A search tool that can quickly extract examples of particular words from the corpus would greatly improve the accuracy of our searches (though spelling errors in the data itself might prevent the searches from being wholly error free). Furthermore, the time taken by the searches would fall dramatically. And crucially, corpora allow access to reliable information regarding *frequency*. In the absence of corpus data, even trained linguists find it very difficult to come up with estimates of frequency in language that are reliable (Alderson 2007).

Being able to search for, and extract frequencies of, different wordforms or phrases gets us a long way. But it does not give us all the tools we need for every sort of research question. We can search for *walking*, *having walked* or *to walk*, but we cannot search for 'every non-finite verb'. Searching for each non-finite form of every verb in English would take a very long time indeed. Quantifying the relative frequency of, say, nouns and adverbs in English would take even longer – to the extent that an investigation of these features based on corpus data is effectively impractical if we use searches based on wordform alone. However, if our corpus already has annotations that show the part of speech of each word in the corpus, then, armed with a search tool that understands these annotations, we are able to fashion a query to extract the information we want – the frequency of nouns, or a concordance of all non-finite verbs – rapidly and reliably. So the combination of corpus, search tool and corpus annotation make it possible to explore research questions that would be almost unimaginable otherwise. As studies such as Leech (2004a) and Mukherjee (2005) show, even with something as apparently mundane as a non-finite verb form, a large-scale investigation can reveal features of its use that have escaped linguists who have used intuition or small numbers of examples alone. Indeed Leech (2004a) is a telling example of this. Between its original publication in 1971 and its third edition in 2004, the increasing availability of corpora and tools to search them has led to parts of the work, especially the chapter on modal verbs, being revised substantially. Two good examples are Leech's (2004a) identification of emergent modal constructions such as *need to* and *had better* and his discussion of *would* as pure hypothesis in expressions such as *I would think* and *one would expect*. In both cases it was the data available to Leech which led to his identification of the growing salience of these features in English usage, and hence to the modification of his earlier work. It is difficult to see how such observations could be made reliably on the basis of any other sort of evidence.[5]

Given that, as discussed in Chapter 1, an ever-increasing amount of corpus material is available for an ever-greater range of languages, the remainder of this chapter will look at a number of issues that impact upon the utility of that data. In doing so we will look at two issues mentioned already – corpus annotation and

Table 2.1 *Metadata stored about two speakers, June and Jonathan, in BNC file KCT*

Name:	June	Jonathan
Sex:	Female	Male
Age:	35–44	0–14
Social class:	C2	C2
Education:	n/a	n/a
First language:	n/a	n/a
Dialect/Accent:	East Anglian	East Anglian
Age:	40	10
Occupation:	dinner lady (pt)	student (state primary)
Role:	self	son

corpus analysis tools – and consider, finally, the statistics commonly used to aid the interpretation of corpus data.

2.3 Corpus annotation

2.3.1 Metadata, markup and annotation

Corpora typically contain within them three types of information that may aid in the investigation of the data in the corpus: metadata, textual markup and linguistic annotation. *Metadata* is information that tells you something about the text itself – for example, in the case of written material, the metadata may tell you who wrote it, when it was published, and what language it is written in. The metadata can be encoded in the corpus text, or held in a separate document or database. *Textual markup* encodes information within the text other than the actual words. For example, in a printed written text, textual markup would typically be used to represent the formatting of the text – such as where italics start and end. In transcribed spoken corpora, the information conveyed by the metadata and textual markup may be very important to the analysis of the transcript. The metadata would typically identify the speakers in the text and give some useful background information on each of them, such as their age and sex. Textual markup would then be used to indicate when each speaker starts to speak and when they finish. Consider the examples in Table 2.1 and Figure 2.1. Table 2.1 gives an extract of the metadata relating to a file in the BNC (file KCT). Using information such as that in Table 2.1, it is possible to limit the searches of the BNC in a way which allows a linguistically motivated question to be posed – for example, to extract all examples of the word *surely* as spoken by females aged between 35 and 44.

Figure 2.1 shows an extract of the orthographic transcription contained within the file itself. The BNC is marked up using a specific convention for encoding

Jonathan	357	It's poached
June		<unclear>
Jonathan	358	Egg in the batter.
June	359	Egg in the batter?
June	360	You <-|-> mean erm <-|->
Jonathan	361	<-|-> cooked <-|->
June	362	scotch egg?
Jonathan	363	Yeah, that's it.
June	364	Well you wouldn't like that surely, cos you don't like sausages.
Jonathan	365	They're not sausages.

Figure 2.1 *A conversation between June and Jonathan (BNC file KCT, utterances 357–365).*

such information reliably – the *eXtensible Markup Language* or XML. This is a standard which is used widely nowadays, not merely for corpus files but also, for example, to transfer webpages and word-processor documents reliably from one machine to another. Figure 2.1 does not show the actual XML codes in the BNC file, which are realised as tags delimited by angle-brackets <like> <this>; rather, the text is formatted in a readable way that visualises the structure of the underlying markup.

Together, where they have been introduced into a corpus, metadata and textual markup allow a range of research questions to be addressed. However, we can go beyond merely recording features of a corpus text such as where italics, or the speech of a certain speaker, begin and end. We can also encode linguistic information within a corpus text in such a way that we can systematically and accurately recover that analysis later; when this is done, the corpus is said to be *analytically* or *linguistically annotated*. Annotation typically uses the same encoding conventions as textual markup; for instance, the angle-bracket tags of XML can easily be used to indicate where a noun phrase begins and ends, with a tag for the start (<np>) and the end (</np>) of a noun phrase:

<np>The cat</np> sat on <np>the mat</np>.

How is such linguistic annotation introduced into a corpus and what are the limits on what may be annotated? There are three approaches to linguistic annotation – purely automatic annotation, automated annotation followed by manual correction and purely manual annotation. None of these approaches is currently error free. Automatic annotation of parts of speech in English, for example, can be undertaken to a high degree of accuracy; Garside and Smith (1997) report accuracy of 97 per cent or more. This means, however, that there are still residual errors. Similarly, with manual processes, it is not possible to guarantee that no errors will be made – no human analyst is perfect. Obviously automatic annotation is desirable where its results are accurate enough – it allows new corpora to be rapidly and cheaply annotated. However, it is not currently possible to reliably undertake automated corpus annotation for all types of linguistic analysis.

Nonetheless, a wide range of annotations have been applied automatically to English text, by analysis software (also called *taggers*) such as:

- constituency parsers such as Fidditch (Hindle 1983), used in the production of the Penn Treebank;
- dependency parsers such as the Constraint Grammar system (Karlsson *et al*. 1995);
- part-of-speech taggers such as CLAWS (Garside *et al*. 1987), used to annotate the BNC;
- semantic taggers such as USAS (Rayson *et al*. 2004), which has been used to annotate a number of corpora (see, e.g., Maclagan *et al*. 2008);
- lemmatisers or morphological stemmers, which are often found as built-in subsystems of many parsers and taggers.

Of course, these automatable analyses may still be applied by hand, or at least manually corrected; and there may often be value in doing so, for instance to create a small, *gold standard* dataset to use as a benchmark for measuring tagger performance. The Manually Annotated Sub-Corpus (MASC; see Ide *et al*. 2010) of the American National Corpus, for example, contains multiple layers of manually inserted or manually checked annotation and is clearly suitable for this purpose.

Other forms of annotation have been applied manually, such as pragmatic annotations (see, e.g., Archer 2005). The range of annotations developed and applied is similar for a number of major languages, such as Chinese, French or German. For a host of other languages, however, the availability of programs which can undertake automatic annotation is patchy at best. For example, there is no publicly available part-of-speech tagger for the Bantu language Kinyarwanda that we are aware of. For such languages, either we must develop an automatic system, perhaps using one that can be retrained to work on a different language, or else we must undertake the analysis by hand.

When a corpus includes linguistic annotations, it is important to note what can and cannot be said about those annotations. Firstly, and perhaps most importantly, we cannot say that the corpus contains *new* information. It clearly does not. What a linguistic analysis of this sort does is to make explicit information that is there implicitly in the data. In other words, identifying a word as a noun does not mean that we transform it into a noun in so doing. In corpus annotation we engage in a process of labelling, not creation or transformation. To that extent, we can say that the corpus is *enriched*, from the point of view of a program or user, but we cannot say that the corpus has had new information added to it.

2.3.2 Consistency of annotation

An important point to make, and this point is vexatious, is that we cannot necessarily say that the analyses encoded in the corpus are *consistent*. If the analysis is manual, and if the annotations were undertaken by a linguist

or linguists working to an agreed set of guidelines for applying the annotation, then we can be much more confident in the consistency of the analysis, which is sometimes measured experimentally using statistics which indicate the degree of inter-annotator agreement (Marcus *et al*. 1993; Voutilainen and Järvinen 1995; Baker 1997). So, for example, the SUSANNE corpus (Sampson 1995) was annotated with part-of-speech and constituent structure analysis following a scheme devised by Sampson. This scheme was based upon his earlier experiences annotating the LOB corpus. The annotations were undertaken according to a strict set of guidelines, subsequently published by Sampson (1995). In this case we have a corpus which exhibits a very high degree of consistency, and we have the means to check that consistency by going back to the published guidelines. However, SUSANNE probably represents a high-water mark for human consistency in manual corpus annotation. In fact, it is inevitable that, from time to time, manual annotations will be inconsistent to some degree. Quite apart from considerations of human error, this is due to a property of all linguistic analyses, namely that an analysis typically represents one choice among a variety of plausible analyses. In the phrase *his future bride*, what part of speech is *future*? It may plausibly be considered a noun or an adjective in this case. To some extent the choice between the two options, while it might conceivably be data- or theory-driven, can be seen as arbitrary – that is, choosing either interpretation is fine, as long as this choice is made consistently. But human analysts are typically not very good at being 100 per cent consistent in such decisions.

Computer programs are rather better at being consistent than human beings – given the same input, a program should theoretically always produce the same output (unless any of the procedures it applies involves randomness, but even then the randomness will be introduced in a consistent way). But even an automated tagger may be inconsistent *over time*. The programs used to annotate texts are from time to time updated or changed in some way. For example, the programmer may change their mind about how they prefer to analyse *future* in *his future bride*, and alter the program accordingly. Even if no such changes of interpretation are made, the output of the tagger will change as the algorithms it uses, or the linguistic knowledge base within it, are extended and improved over time. So inconsistency may be inevitable in corpus annotation, whether manual or automatic. But this cannot be seen as a major objection to the practice of corpus annotation because, as we have seen, inconsistency is inevitable in linguistic analysis in general. The virtue of corpus annotation is that, because the choice of the analyst (or annotation software) is explicitly present in the text, any inconsistency is clear and open to scrutiny, and any necessary allowances can be made by users of the data. Without corpus annotation, the potential for inconsistency in analysis still exists, but the prospect of being able to discover it in the work of an analyst is at best remote. This advantage of corpus annotation becomes important when we consider the replicability of our research (see section 1.6.1). A finding based on analyses explicitly available as annotations in a text is fundamentally more

easily replicable than a finding based on analyses of which other researchers do not have a record.

Inconsistency exists at another level in corpus annotation – the consistency of annotations between corpora. Given a scheme for part-of-speech tagging, a team of researchers might work to develop a corpus annotated consistently using that scheme. But such a project would use only one of potentially many schemes for introducing part-of-speech information into corpus data. This is not necessarily a problem – one scheme may simply be more detailed than another. But schemes of analysis can vary in other ways. For instance, linguists may have different views as to what parts of speech exist in a specific language. Even looking at just one language, the same analysts may change their views on what annotation scheme they should use. Over the years, seven different part-of-speech annotation schemes have been developed and applied by the team behind the CLAWS tagger (Garside *et al*. 1987). This variability in annotation schemes can become a problem for users trying to use multiple annotated corpora at once. If the annotation scheme is not consistent across the corpora, then, at best, a translation from one scheme to the other will be needed; at worst, such a translation may not be possible. This problem, though a subject of research for some time (Atwell *et al*. 1994), has come to the forefront for users of annotated corpus data, notably computational linguists (see Chapter 9); and much effort has been expended to develop ways of using corpora annotated with different annotation schemes (see Meyers 2009 for a detailed discussion). Early indications are that the problem is not insurmountable, though, as Meyers (2009: 122) notes, 'a greater degree of coordination among annotation research groups would vastly improve the utility and accuracy of annotation'. This has been tried to some extent in the past, with the Expert Advisory Group on Language Engineering Standards (EAGLES), which worked in the 1990s, being the most prominent example to date.[6] However, no matter how admirable the goal, the ability of linguists to quite legitimately disagree when they are working out schemes for annotating language data means that the labours of researchers like Meyers are likely to be difficult at best and Sisyphean at worst.

Automated corpus annotation is currently conceived of as a distinct type of corpus processing, treated independently of tools for searching a corpus. The overall procedure thus typically has two steps. Firstly, the untagged text of a corpus is loaded into the annotation tool and tagged, producing a version of the text with tags encoded into it using XML or some other formalism. Then, this annotated corpus is loaded into a separate search tool so the researcher can enter their searches and get back the results. These two steps are usually implemented separately, for mainly practical reasons. As we will explain in section 2.5, corpus search tools have over the years developed to become very user-friendly. By contrast, corpus annotation programs – while widely available – typically require so much advanced computer expertise to install and use that they are, effectively, not accessible to most linguists.[7] Moreover, annotating a text is typically a much

longer process than searching it – while nowadays search software can often produce results at a speed that appears pretty much instantaneous to a human being, annotation software is computationally intensive and can take hours or longer to run. So frequently a corpus will be tagged 'once-and-for-all', often by its builders, and the tagged text passed on to users who run searches on it.

There is no reason why tagging and searching have to be separate in this way, however. Let us consider a search for a part-of-speech category such as *noun*. To accomplish this with automatic annotation, the computer has to firstly decide which words in the corpus are nouns, and secondly present all the words that it has identified as nouns. It is entirely possible for these procedures to be carried out at the same time – in which case the intermediate step, where a corpus with annotations encoded into it is saved to disk and then transferred to a different tool, becomes completely unnecessary. There already exist tools which bring together annotation and search under a single roof, such as GATE,[8] Wmatrix (Rayson 2008), and SketchEngine (these latter two tools will be discussed in detail later on). In these cases, however, the tagging and searching systems are still independent, behind the scenes. More notably, the Nooj system (Koeva *et al*. 2007) is a complex, multilingual annotator-cum-concordancer in which the procedure for annotating some linguistic feature and the procedure for searching for it are one and the same.

We may expect that, in the future, joint annotation–search tools of both these kinds will become more common. In many ways this development is much to be desired, as it will open up automated annotation to users of corpus data who could not hope to manage the technicalities of much tagging software. It will also free users from the strictures of the annotation applied to a corpus by its builders, allowing them to choose what annotations they want to apply. There is, however, also a downside to this kind of on-the-fly annotation. Firstly, it removes the possibility of running manual checks on the output of an automated tagger before using it, whereas this is very easy when annotation and search are firmly separated. While this is not an unimportant consideration, it is in practice rare for such checks to be carried out in studies where the annotation is ad hoc, rather than part of the preparation for releasing a dataset publicly. In most such studies, the published error rates for a tagger are simply accepted and allowed for, so in these cases nothing would be lost by joint annotation–search. Secondly, and more seriously, if no annotated version of the corpus is actually created on disk and archived, then the advantages of corpus annotation with regard to replicability that we discussed above are undermined. Tools such as Wmatrix, where a copy of the text with encoded annotations *is* created in the background and is available to the user, may therefore be preferable to combined annotation–search tools that do not have this feature.

Combined annotation–search tools may be the direction of the future, but today we must deal in large part with tools that are designed for searching alone. A range of such tools is vitally necessary to explore a corpus in service of our research question. A well-annotated corpus may be of inestimable value, but

把 他 打 得 **晕头转向** ， 他 不得已 **报**	**了**	**师范** 院校 ， 尽管 **录取** 他 的 **师范大学** 依
觉得 自己 已 从 宝塔 **顶** 上 跌落 下来	**了**	。 入学 后 ， 肖 立 从不 佩戴 校徽 ， 未来
不 知道 ， 儿子 的 枕 下 已 放 **满**	**了**	武侠小 **说** 、 黄色 **书刊** 。 直到 大学 四 年**级**
， 四 门 功**课** 不 及格 的 **现实** 一下 **压垮**	**了**	肖 立 。 茫然 中 ， 他 想到 死 。 在 死
欢乐 ， 于是 ， 他 魔鬼 般 地 扑 向	**了**	**邻居** 的 一个 女孩子 … … 他 在 惊慌 中
子 … … 他 在 惊慌 中 残忍 地 打 昏	**了**	女 孩儿 ， 自己 跌跌撞撞 地 逃 出 了 小
了 女 孩儿 ， 自己 跌跌撞撞 地 逃 出	**了**	小屋 … … " 当 我 打 那 女孩 儿**时** ， 我
可是 ， 她 没 昏 … … 我 慌	**了**	， 束手无策 … … 那 一会儿 ， 我 **脑子** 里
第二 次 **谈话 时** ， 我 和 他 一起 分析	**了**	他 犯罪 的 原因 。 他 **说** ： " 如果 可能

Figure 2.2 *An extract of a sample concordance of the particle 了 from the LCMC.*

realising that value is dependent on being able to search it in a way that is reliable and linguistically motivated. So what tools of this kind have been developed, and what kinds of analyses do they enable?

2.4 Introducing concordances

Undoubtedly the single most important tool available to the corpus linguist is the concordancer. A concordancer allows us to search a corpus and retrieve from it a specific sequence of characters of any length – perhaps a word, part of a word, or a phrase. This is then displayed, typically in one-example-per-line format, as an output where the context before and after each example can be clearly seen. Figure 2.2 shows a concordance for the Chinese word *le* (了), taken from the Lancaster Corpus of Mandarin Chinese (the LCMC; McEnery and Xiao 2004a) using the CQPweb concordancer (Hardie forthcoming).[9]

We have avoided saying that a concordance shows *words* in their context – though this procedure is often called *key word in context* (KWIC) concordancing, KWIC need not be limited just to showing whole words. For example, in English we might want to produce a concordance of a common suffix in order to explore its context of use – Figure 2.3 shows an example of this, a concordance of words ending in the nominalising suffix *-ness*.

Likewise, if we have an interest in idioms, we may wish to view a concordance of a multi-word expression. Importantly, if there is some form of linguistic annotation embedded within the corpus data, we may wish also to search not for a linguistic form at all, but rather for one of the annotation mnemonics – so, for example, a search for words marked with the part-of-speech tag *u* will return all of the 75,273 auxiliaries within the LCMC.

Exeter : the capital city of	**blandness**	ca n't break the chain TWO years ago the
after Yahoo warned it was seeing	**weakness**	in two of its biggest advertising segments .
advertising in the face of economic	**weakness**	. '' The warning , which was similar to
in the afternoon following news of more	**weakness**	in the US housing market , with housing
became a lifeline for Ross when his	**illness**	kept him confined to hospital and his home
Olympic bid . She was briefly married to	**fitness**	guru John Crisp . Her second marriage to
and are far less secure . " Ironically , this	**hyper-attentive-ness**	is actually having a damaging effect on our
thing is absolutely barmy . " And the	**madness**	continues with the goalposts constantly
notes led to a lifetime of passion and	**happiness**	. She was just 17 and he was 18 when they
so much , just to feel your warmth and	**warmness**	around me (lovely , lovely thought) . I 'm
Capt Alfred Bland MY only and eternal	**blessedness**	, I am never utterly miserable , not even
will and giving up a chance of supreme	**happiness**	with you . What more can anyone ask ?
made me during those days-just you , my	**sweetness**	. MY darling ... the nights here are weird .
range hens under threat as devastating	**illness**	nears Britain POULTRY farmers in
and has submitted it for inclusion in the	**Guinness**	Book of Records . 'Unfortunately , the

Figure 2.3 *An extract of a concordance of words ending in* -ness *from BE06 (Baker 2009).*

Corpus search tools are vital if the range of research questions we can address using a corpus is to be significantly expanded. But while concordancers, and other tools for exploring a corpus, are powerful aids to the linguist, they also, crucially, limit and define what we can do with a corpus. On the other hand, it is for practical reasons impossible to avoid using these tools. The analysis of very large corpora without computer processing can best be regarded as what Abercrombie (1965) referred to as a *pseudo-procedure* – a method in linguistics that might yield very interesting results and be useful in principle but which is, for all practical purposes, impossible. It is simply too time-consuming. In fact, fundamentally, the corpus-based approach to language cannot do without powerful searching software. Conversely, concordancers of suitable power remove the pseudo-procedure argument for that set of research questions which a concordancer, working in concert with a specific corpus, can address. For example, consider the LCMC. This is a million words of Mandarin Chinese from the early 1990s which has been annotated with part-of-speech information. The corpus was collected according to the LOB sampling frame. There is a range of questions which can be addressed using this corpus; equally there is a set of questions that cannot be answered readily, or even at all, using the corpus. Some of these questions are excluded by virtue of the composition of the corpus; it contains no nineteenth-century texts, so it cannot be used to explore the Mandarin of that period. However, some questions are excluded by virtue of the available annotation. While we can use the part-of-speech data in the corpus to explore various research questions (many relating to Mandarin grammar), there are, equally, questions that we cannot answer because the annotation we would need to perform the necessary queries is not present or

not searchable. For example, there is no annotation of constituent structures in the LCMC. In the absence of an automatic phrase-structure parser, and a tool to search its output, the use of the corpus to study constituent structure in Mandarin may well still be considered a pseudo-procedure. Of course, we might approach constituent structure indirectly by means of searches for words or part-of-speech tags. But it is impossible to *directly* address this research area without the appropriate annotation. This is an important point; the corpus alone solves few (if any) problems for a linguist. Its potential is unlocked by tools that allow linguists to manipulate and interrogate the corpus data in linguistically meaningful ways. The availability of tools that are relevant to specific research questions remains a crucial limiting factor in corpus linguistics. Over time, an increasing number of tools *have* become available, and this has expanded the range of research questions that may be addressed using a corpus. But a range of research questions do still lie beyond the reach of what can be done with a corpus at the present time. We will return to this topic in section 2.5.4.

2.5 A historical overview of corpus analysis tools

When did the process of developing software tools for corpus analysis begin? The honour of doing this may reasonably be claimed by Roberto Busa, who built the first machine-readable corpora and undertook the first automated concordances in 1951. Busa did not, however, invent the concordance; although it was in the realm of the pseudo-procedure for most purposes, some hand-compiled concordances had been available for some key works for a long time. For example, Hugh of St Cher, with the assistance of around five hundred monks, compiled the first concordance of the Latin Vulgate Bible in 1230, providing, for each word, an index of where each instance of it could be found. However, Busa showed that, with a little effort, concordancing could be applied rapidly and effectively to electronic texts. This was a pivotal moment, when concordancing moved from being a labour-intensive task applied to a few texts of particular cultural importance – such as the Bible, the Qur'ān, or the works of Shakespeare – to being a technique that could, in principle, be applied to any text at all. Busa's work led to what we will term *first-generation* concordancers.

2.5.1 First-generation concordancers

First-generation concordancers were typically held on a mainframe computer and used at a single site, such as the CLOC (Reed 1978) concordance package used at the University of Birmingham.[10] Individual research teams would build their own concordance system and use it on the data they had access to locally. The packages typically did no more than provide a KWIC concordance. Any other manipulation of the data was undertaken by separate programs. So, for

example, to build a list of the words appearing in a corpus, a different program would be used, such as that used by Hofland and Johansson (1982) to produce word frequency lists of English. First-generation concordancers typically had great difficulty dealing with anything other than the non-accented characters of the Roman alphabet – where characters carrying diacritics appeared in English texts, for example, they would be replaced by character sequences designated to represent those characters. These character sequences were usually agreed on a site-by-site basis. For example, in LOB, an apostrophe after a vowel indicates an acute accent on the preceding letter – so the word *café* would be encoded as *cafe'*. (More recently, standardised sequences such as *café* in which the XML code *é* replaces the <*é*> character have been used.) This points to a general limiting factor that runs through the history of concordance software. Where an international standard did not exist for some feature of corpus use or storage that had to be dealt with, concordance program writers and corpus builders improvised a way around the issue. Another example of this relates to how markup and annotation were encoded in corpora. Standards such as XML (see section 2.3.1) have now emerged to allow information such as linguistic annotations to be reliably inserted into corpus texts. But no such standards existed to begin with. Accordingly, as with character encoding, ad hoc standards emerged in those centres developing corpus texts and software tools. So, for example, at Lancaster in early corpora such as LOB (Johansson *et al*. 1978) and the Spoken English Corpus (Knowles 1993), an underscore character was used to associate a word with its part of speech. So an instance of the word *dog* acting as a noun in the corpus would appear as *dog_NN1*, where NN1 is the mnemonic for a singular common noun. Tools which manipulated these corpora, such as Roger Garside's XANADU (Fligelstone 1992), were programmed to interpret annotations of this sort. First-generation concordancers were useful in development terms as they clearly showed the need for standards – if corpora were to be passed between users at different sites, then some agreement had to be reached on how this kind of information was to be encoded. They also showed how wasteful it was for people to reinvent the wheel. It was clearly preferable to write concordancers that could work on a range of machines and a range of corpora. There was a further clear advantage to doing this: replicability (see section 1.6.1) was difficult to achieve in an era when, though corpora were shared, the tools for manipulating them were not. While it might be hoped that results would be broadly replicable between sites, without both the corpora and the tools on which a study was based being made widely available, it was difficult to test the replicability of any of the results produced at a site without visiting that site and getting permission to work there.

First-generation concordancers also clearly demonstrated that, rather than having stand-alone tools for functions like generating frequency lists, it was preferable to write software which bundled many tools together into a single package which, while it might still be called a concordancer, would allow the data to be manipulated in as wide a range of ways as possible. The widespread

availability of personal computers from the 1980s onwards removed a major barrier to solving these problems. A problem with mainframe computers was that programs written to run on one mainframe would often not run on a mainframe of another type. While some effort might make the program work, that effort might be so great that just starting from scratch and writing your own KWIC concordancer might seem preferable.

2.5.2 Second-generation concordancers

Second-generation concordancers were enabled by the spread of machines of one type in particular across the planet – IBM-compatible PCs. It became possible to write and distribute a concordancer for that platform with a high likelihood that the program would be usable straight away on the recipient's machine. This was important in two ways. Firstly, it meant that a lot of effort that had gone into reinventing the wheel could now be directed towards producing better tools. Secondly, it had a democratising effect. Up to this point corpus linguists typically needed to work in a team which included a computer scientist who was prepared to do whatever programming was needed, either to produce tools or to make somebody else's tools work on the local mainframe. With PC-based concordancing, any linguist who was able to switch on and use a PC could use corpora. The effect was fairly immediate. Corpus linguistics boomed from the late 1980s onwards. The increasing availability of tools such as the Kaye concordancer (Kaye 1990), the Longman Mini-Concordancer (Chandler 1989) and Micro-OCP (Hockey 1988) enabled this boom. Second-generation concordancers, however, inherited many of the problems of the first generation. They provided very few tools other than KWIC concordancing. They could typically sort alphabetically the left and right context of the word searched for, produce a wordlist, and calculate some basic descriptive statistics about the corpus (word count and type–token ratio, for example). They also had the same issues with character representation and corpus formatting as the first-generation concordancers – standards had not developed to the point where generally agreed formats could be relied upon.

Partly as a result of this, the second-generation concordancers were, arguably, worse than the first generation. The second-generation concordancers generally made no assumptions about a corpus being marked up in a particular way. So, for example, when using the Longman Mini-Concordancer with a corpus marked up in the Lancaster convention of *word_TAG*, the user had to be aware of and understand that convention, and develop searches accordingly. For instance, a search for *dog* would find nothing if all instances in the corpus appeared as *dog_NN1*! The computer just interpreted the corpus as a very long stream of undifferentiated text – it did not 'know' what parts were words and what parts were part-of-speech mnemonics. By contrast, the earlier bespoke programs written for a specific group were often able to prise the two apart. First-generation concordancers were also superior to the second generation in terms of *scale*, that

is, the size of the corpora they could successfully search through. For instance, because of the limitations of early PCs, the Longman Mini-Concordancer could search through a few tens of thousands of words before it ran out of memory. Meanwhile, at that point in history, tools running on mainframes were able to deal with corpora of a million words or more, albeit slowly at times. However, the energy released by ending the need to reinvent the wheel, and the continuing increase in the power of PCs, led to a third generation of concordancers.

2.5.3 Third-generation concordancers

The third generation of concordance software includes such well-known systems as WordSmith (Scott 1996), MonoConc (Barlow 2000), AntConc (Anthony 2005) and Xaira.[11] These concordancers were able to deal with large datasets on the PC (the 100-million-word BNC is packaged with Xaira). Moreover, they had bundled in with them a wider range of tools than had previously been available in concordance packages, and they gave access to some meaningful statistical analyses which went beyond the merely descriptive. Finally, they effectively supported a range of writing systems.

The last point deserves a little more discussion. Stable and widely accepted standards for encoding information in corpora – notably XML – have developed in recent years, as discussed previously. At the same time, the encoding of the actual characters in a corpus has also standardised around a system called Unicode. Before Unicode it was very difficult for a system developed to support the Roman alphabet to read, appropriately display and effectively analyse corpora in other writing systems. With the advent of Unicode, computers and the programs that run on them have become much more effective at dealing with data in a range of writing systems. Third-generation concordancers have developed to exploit this new capability – for instance WordSmith, as of version 4, can be used to analyse Unicode-compliant corpora. This represented a great breakthrough for corpus linguistics and, if nothing else, has saved a great deal of effort. Prior to Unicode being widely implemented, concordancers had to be developed to cover each writing system separately, leading to the production of concordancers that could only be used with certain scripts such as Arabic (Abbès and Dichy 2008) or Chinese (Luk 1994). Where concordancers were developed that could be used with a range of scripts (Wools 1998), effort and some technical expertise was required by the user to retarget the concordancer at a corpus in a new writing system. Unicode has removed the need to redevelop basic packages to support concordancing across writing systems. In so doing, it has greatly simplified the task of working with a disparate range of languages and has increased the ease with which corpus linguistics can be adopted. As more and more Unicode corpora become available, the usefulness of the Unicode-compliant concordancers will become ever more apparent.

It is notable that the third generation of concordancers are in many ways remarkably similar to each other, especially in terms of their core functionality: concordances, frequency lists, collocations[12] and keyword analysis[13] are the main tools available in each. The ubiquity of these four analysis functions is not surprising. Given a corpus of texts, a computer can do two basic things: count the things in the corpus (a frequency list) and locate all of the examples of a search term and display them (a concordance). It can then derive statistical abstractions on each of these outputs: keywords are a statistical abstraction from frequency lists and collocations are a statistical abstraction from a concordance. Each of these procedures may then be carried out on annotations as well as words (so, we can get a frequency list of part-of-speech tags, or calculate key tags, or calculate which tags collocate with a particular search term), although the degree to which tools allow annotations to be referenced independently of words can vary. Beyond these core procedures, no others have yet emerged as widely agreed to be essential for a concordancer. In short, then, the set of functions which analysts are using to explore corpora seems to be relatively stable at present. There are differences in the specialisations of particular tools, of course. Xaira, for instance, does not include any tool for the calculation of keywords, a core function of WordSmith and AntConc; but it does provide powerful support for searching for XML elements (whether markup, annotation or metadata). Tools also differ in the extent to which they support lists of sequences of words in a corpus (*n-grams*). Yet at their core the third-generation packages are, we would argue of necessity, similar.

Given that the tools embedded in these programs were largely available during the first generation of concordancing – albeit to a much smaller audience – should we conclude that corpus exploitation techniques reached such a level of maturity relatively early on that all that is left to do at present is to repackage these tools, and perhaps make them easier to use? Almost certainly not. Three pieces of evidence tell against this conclusion. Firstly, there are tools from the first generation of concordancing that are not generally available in third-generation concordance packages. A good example of this is the technique of *collocational networks* developed by Phillips (1989). This method appears to be useful, and has been used since Phillips developed it, but always in the form of bespoke software which is neither embedded in popular concordance packages nor publicly available (see McEnery 2005: 20–3). Collocational networks provide a telling example of how the inventory of tools developed in the first phase of corpus software development has yet to be transferred in full into the third phase. Another example of the same phenomenon is the multi-dimensional (MD) approach of Biber (1988; see section 5.4). MD is a highly influential approach to the analysis of corpus data; yet although the general thrust of the approach has been roughly duplicated using general corpus searching software (Tribble 1999; Xiao and McEnery 2005), there is no easy-to-use integrated package publicly available that will perform a full MD analysis from beginning to end in a sufficiently user-friendly way to make it accessible to the majority of linguists. Secondly, as we can see from

the current literature, there are techniques that have been developed recently which are not incorporated in contemporary concordancers. A good example of this is collostructions (Stefanowitsch and Gries 2003; see section 7.5.1). These are potentially important to discover in corpora, yet third-generation packages such as WordSmith and AntConc do not directly support users in extracting them. As with MD, with a little ingenuity some existing tools, notably collocation packages, can be used to begin the process of finding collostructions, but this falls short of the necessary full support for discovering them with accuracy, speed and ease. Finally, there are specialised concordancers which, while lacking the range of features available in popular third-generation concordancers such as WordSmith, nonetheless provide tools which are of clear relevance and importance to linguists. Probably the most striking case of this is the ICECUP program (Quinn and Porter 1996) provided to support the analysis of ICE-GB (the British component of the International Corpus of English).[14] ICE-GB has been annotated with syntactic tree structures; it is thus a so-called *treebanked* corpus. In a treebank, the main syntactic constituents of each sentence are annotated; one common system of annotation uses brackets to which a mnemonic tag is added to denote the constituent (S for simple declarative clause, NP for noun phrase, PP for prepositional phrase, VP for verb phrase, etc.). The following example from a treebanked corpus using this format is drawn from Taylor *et al.* (2003: 7):

> ((S (NP Martin Marietta Corp.) was (VP given (NP a $29.9 million Air Force contract (PP for (NP low-altitude navigation and targeting equipment))))).)

Treebank annotation is typically very complex and multilayered – the parsing structures embedded in ICE-GB are considerably more detailed than the example above. The ability to search a treebank rapidly and effectively is clearly a significant advantage, as is the ability to get clear graphical displays of the output, since the underlying bracketed text gets harder and harder to read as the analysis gets more detailed. ICECUP has both these capabilities. As such it is a highly useful and sophisticated piece of software, despite the significant drawback that it can only be used with the style of syntactic annotation employed by the team of linguists who created ICE-GB. Yet this search and display functionality is currently lacking from concordancers in general use. In this case we might be able to address the shortcomings of the general packages by switching out to a specialised program such as ICECUP to carry out relevant searches where necessary. But at best this can be described as a sub-optimal solution, and it is probably better characterised as quite unsatisfactory.

In short, there is plentiful evidence – from tools developed in the past and in use at present – that the range of the existing corpus analysis software could very usefully be extended, either in terms of user-friendliness or function. This is clearly an important issue in corpus linguistics; as we have noted, if the toolset does not expand, then neither will the range of research questions that may reasonably be addressed using a corpus. We might ask *why* general corpus search

tools do not incorporate these kinds of extended analyses. The answer may be different for each type of analysis, but in general it is related to the limited resources of developer time available for the creation of a software tool. The third-generation concordancers were based largely upon the efforts of talented and committed individuals – for example, AntConc was developed by Laurence Anthony and WordSmith was developed by Mike Scott, each working alone. While both were assisted to a degree by enthusiastic users who helped them to debug and expand the functionality of their programs, they laboured on their packages with no external aid, relying instead on their considerable talent and deep commitment to helping others. While this is entirely noble and admirable, there are also hard limits to the amount of time available to individuals working in this way. This being the case, it is unsurprising that they have concentrated on implementing the corpus analyses – concordances, collocations, keywords – that are of the greatest generality and of use to the most researchers. Many of the extended analyses we have discussed are restricted in their application for either practical or theoretical reasons. For example, as noted, ICECUP only works with corpora annotated in the style of parsing used at the Survey of English Usage (SEU) research unit at UCL. It cannot be used to search any other parsed corpus (or, indeed, corpora with other kinds of annotation). But there is a very wide variety of parsing schemes, of which the SEU's system is only one. This explains to a large extent why no ICECUP-like search and display system is found in any general tool: the tool's author would have had to expend a great deal of effort to implement this functionality, for the benefit of a relatively narrow subset of the users – those using corpora with SEU-style syntactic annotation such as ICE. Similarly, collostructions are associated with a particular theoretical approach to language (namely Construction Grammar). So implementing direct support for collostruction searches in a general corpus package would only be of help to users working within theoretical frameworks compatible with Construction Grammar. The limitations of the general software are, then, quite understandable. Nonetheless, it is still deeply regrettable that the basic toolset places these implicit restrictions on the expansion of the field of corpus linguistics.

2.5.4 Fourth-generation concordancers

Considering the need to expand the range of corpus analysis tools that are available, it is a pity to note that the fourth generation of concordancers are strikingly similar, in terms of their functionality, to the third. In fact, fourth-generation concordancers have arisen, not to extend the range of available analyses but to address three entirely different issues: the limited power of desktop PCs, problems arising from non-compatible PC operating systems and legal restrictions on the distribution of corpora. We will discuss the legal issues associated with corpus construction in the next chapter. However, it is often the case that a corpus, once collected, cannot simply be given away to any researcher who wants to use it, because that would violate the original text producers'

copyright in the texts within the corpus. This is a deeply frustrating restriction for the corpus builder; it means that fewer people will be able to take advantage of their hard work, and it also reduces the scope for replicating the results achieved with one particular corpus. In the past, corpus builders would get around this issue by distributing the corpus only to institutions who could officially sign up to a restrictive licence, or simply by not distributing the corpus. However, a preferred solution more recently has been to make the corpus available through a web-based interface. That is, a website is created where users can enter search queries and get back dynamically generated results. The proportion of any given text contained within a concordance line is usually no more than a short chunk of a sentence. This is typically thought to fall within the level of 'fair use' allowed under copyright law, and thus does not violate the rights of the text producer (although the legality of such 'fair use' redistribution has yet to be comprehensively tested). For example, Mark Davies made the BNC available via such a website; the Polish PELCRA corpus and the Hellenic National Corpus are examples of corpora whose *primary* public availability is via a web-search interface.[15] These interfaces were not only motivated by legal considerations, of course. They also allowed corpus builders to make their work available immediately, and via a piece of software (the web browser) that all computer users are already familiar with. By being available across the web, they were instantly available to users on any operating system – in contrast to third-generation tools such as WordSmith, only available on Microsoft Windows, or Xkwic, only available on Unix-like systems. Finally, these web-based systems are typically capable of much faster searches than a PC-based concordancer. The corpora to which they offer access are often very large, on the order of a 100 million words. Searching such a corpus on a desktop PC can be a very lengthy process; while modern PCs are very much more powerful than those that the Longman Mini-Concordancer was developed for, any corpus software designed for a PC is still subject to the limitations of memory and processing power of a given user's computer. But by using the web, the fourth-generation tools have effectively decoupled local processing power from the issue of corpus searching. Searches can be completed much faster than the same searches would be if running on the desktop. Thus, these interfaces combine the advantages of the PC and 'mainframe' approaches – users are given access through their browser to a distant, sometimes more powerful, machine on which the actual corpora and corpus processing tools are held. This is known technically as a client/server model – the PC has on it a 'client' program, mediating between the user and the machine on which the real analysis of the data takes place, the 'server'. The client passes requests from the user to the server, and relays back to the user the results of the server's work.

We consider the use of this client/server model, via the specific medium of the World Wide Web, to be the defining feature of fourth-generation concordance tools. Client/server systems already existed in the third generation. Xaira, for example, consists of a server program and a client program (they often run on the

same machine, but do not have to). Xkwic is another example of this arrangement. But it is the move onto the web that frees the tool from the restrictions of local processing power and makes the user's operating system irrelevant. In web-based concordancers, the only thing that is done locally is the rendering of the webpage that contains the results. Numerous web browsers exist for all operating systems, some commercially developed and some not; most computers now come with at least one pre-installed. So the software on the client end can be taken as read – it is only the software on the server end which needs to be developed. The server software normally consists of a powerful and complex search system, which usually works on a corpus that has been *indexed* in some way. Indexing refers to any processing of the corpus which allows words, phrases or tags to be looked up in a search *without* the program going through the entire corpus in search of them. Indexing is crucial if very large corpora (tens or hundreds of millions of words) are to be searched at an acceptable speed. The details are highly technical and vary for each system. But in general, there are two types of software used at the server end. The first is a database system. If a corpus is loaded into a standard database management program, then concordances can be carried out very efficiently using the normal query language of that database system. Since databases are an important application of information technology in general, extremely powerful database software has been developed over the years. Most databases use a formalism called the Structured Query Language or SQL. Examples of SQL-based web interfaces include the already-mentioned PELCRA reference corpus of Polish and Mark Davies' corpus.byu.edu interface (Davies 2005, 2009a).[16] The other type of software that can be used is a dedicated corpus indexing and querying system. The mostly widely used of these is CQP, the Corpus Query Processor, which is a part of the Open Corpus Workbench (CWB),[17] developed at the Institut für Maschinelle Sprachverarbeitung at the University of Stuttgart (see Christ 1994). CQP is the server program behind the Xkwic concordancer. However, it is now more often used as a back-end to web-based interfaces.[18] The Xaira server program can also be used as a website back-end.

While fourth-generation corpus analysis tools began as websites allowing the searching of specific corpora, they have now been extended beyond this into generalisable systems. Three of these are of particular note. The system developed for the BNC by Mark Davies, mentioned above, has been extended by him to allow access to a wide range of very large corpora via his corpus.byu.edu site. This tool is now effectively non-corpus-specific. It is probably the most powerful, and most widely used, SQL-based corpus analysis tool at the time of writing. Another important system is SketchEngine, which uses a CWB/CQP-compatible program called Manatee as its back-end and allows users to analyse a wide range of corpora for lexical and lexicogrammatical patterns; SketchEngine is particularly useful in support of corpus-based lexicography (Kilgarriff *et al*. 2004). Finally there is BNCweb (Hoffmann *et al*. 2008) and its clone CQPweb (Hardie forthcoming),

which combine an SQL database with a CQP back-end. As with Davies' (2009a) system, the original BNCweb is an interface just to one corpus (the BNC), and its re-engineering as a corpus-independent system, CQPweb, was a later development. But as noted above, these systems have very similar functionality to third-generation software such as WordSmith, though they have made progress by adding support for annotation and for regular expressions,[19] which allow more complex patterns to be retrieved from the corpus.

Fourth-generation corpus analysis tools are even more user-friendly and powerful than the third-generation tools (although they are typically not as useful in the study of very small corpora or individual texts). But they do not noticeably expand the range of analysis tools available within general purpose packages. How can this be addressed? If the market for corpus processing software expands sufficiently, it is not inconceivable that a major software company might develop a corpus processing system. Teams of professional programmers could easily produce a system to surpass what was possible for the pioneering individual developers of the third-generation systems. However, to date no major software company has undertaken such an enterprise. Instead, the pioneers of the third- and fourth-generation concordancers continue to push on. One promising approach is open-source modular concordancers (Anthony 2009). The idea here is that anyone working with corpora could create a tool within the framework of an existing concordance program whose source code is openly available. The new tool is then released to all users of the original system. This reduces the amount of work that has to be done to get the new tool up and running, and means that less reinvention of the wheel is done. If this approach was adopted in the context of fourth-generation concordancers, then a comprehensive solution to the problems facing corpus tools developers and users could well emerge. However, there are certain practical difficulties. Most corpus search and retrieval programs are hugely complicated systems, so complicated that it can be very difficult for anyone other than their primary maintainer to understand their internal workings well enough to begin to program extensions. For instance, while both Xaira and CWB have been open-source software for several years at the time of writing, the overwhelming majority of the development work is done by one or two people in each case. As an alternative to the idea of an open-source, modular approach, it has been suggested that corpus linguists, rather than using general corpus analysis packages, should instead fully embrace computer programming and individually develop their own ad hoc tools to address the tasks that face them. This view is probably most forcefully put by Biber *et al*. (1998: 254) who see the following advantages to this approach:

> concordancing packages are very constrained with respect to the kinds of analyses they can do, the type of output they give, and, in many cases, even the size of the corpus that can be analyzed . . . Computers are capable of much more complex and varied analyses than these packages allow, but to take full advantage of a computer's capability, a researcher needs to know how to write programs.

In particular they see the key advantages of this approach to be that (Biber *et al*. 1998: 256):

- you can do analyses that are not possible with concordancers;
- you can do analyses 'more quickly and more accurately';
- you can tailor the output to fit your own research needs;
- you can analyse a corpus of any size.

Each of these advantages is potentially significant where this approach may be taken. However, there are notable limits to this approach:

1. Not all linguists want to be computer programmers – second-generation concordancers promoted corpus linguistics by providing tools to the significant majority of linguists who may want to use corpora without becoming computer scientists *manqués*.
2. Small teams or individuals working alone can only go so far. By coordinating the work of teams towards a common goal we might reasonably expect greater progress in corpus tool development to be made.
3. By working together on corpus and tools development, the process of the development of corpus encoding standards was accelerated. Without this impetus to develop standards, it is unlikely corpora would be as widely used as they are today, as they would be mired in conflicting formats with tools being developed and redeveloped to deal with those formats.
4. Locally developed tools need to be made available to other researchers so that findings based upon them may be replicated and verified. Where tools are made widely available by developers, the user of the tools is not faced with this demand.

However, Biber *et al*.'s points are not necessarily in conflict with the development of general purpose corpus analysis packages. It is often by teams, especially interdisciplinary teams with expertise in linguistics, programming and statistics, working in the framework that Biber has suggested, that new tools are developed in the first place – only later finding their way into concordance packages. Similarly, in Anthony's vision of a cooperative approach to corpus software development, the work of isolated linguists building tools is integrated into widely available packages; in this way, the efforts of those linguists who wish to undertake computer programming, or who collaborate with computer programmers, could be made available to the benefit of all linguists. In that spirit, it is a good idea indeed for those linguists who can do so to continue to develop new tools for corpus exploitation. There is certainly more support available to them through books such as Mason (2001), Gries (2009a) and Weisser (2009) than there has ever been before. Yet for those who cannot or do not wish to program, the fruitful avenue of cross-disciplinary cooperation is still open.

Before leaving the question of the future development of corpus processing tools, however, it should be remembered that if there are problems in corpus processing to be addressed in the future, dealing with those problems is now possible because some basic problems of the past have been addressed. An accepted standard now exists, in the form of Unicode, for encoding a range of writing systems. Likewise generally accepted standards for formatting information in corpora now exist, such as XML. The utility of annotating linguistic analyses in corpora is widely accepted, albeit not universally. Corpora and corpus processing tools are now available virtually to all, not just to a small number of well-funded groups. All of these very real developments have allowed corpus linguists to look to a future where further tools, of great utility to all languages and the majority of linguists, might be realisable. These may include concordancers which are just as useful when dealing with audio and video recordings as they are when dealing with textual material.[20] Allowing for a clearer flow of information between concordancers and database and spreadsheet programs may also greatly facilitate the ease with which corpus data may be manipulated.

2.6 Statistics in corpus linguistics

Corpora are an unparalleled source of quantitative data for linguists. It is hardly surprising, therefore, that corpus linguists often test or summarise their quantitative findings through statistics. It is possible to use a corpus and, quite legitimately, not engage in statistical analyses. For example, to establish that a particular phoneme *exists* in a given language, all we have to do is find one good, well-attested example of that phoneme occurring. This is an all-or-nothing assessment – finding a thousand examples does not necessarily help us any more than finding one. Of course, *not* finding an example does not necessarily count as evidence of non-existence (though it may count as evidence of rarity).

However, if we want to establish that the same phoneme is *frequent* in that language, we need to employ statistics, because frequency is not an all-or-nothing matter. In this case, we would need some basis of comparison on which to assert that the phoneme is frequent. Frequency cannot be measured in an absolute sense such that *frequent* has an invariable value associated with it (saying flatly 'anything above twenty counts as "frequent"!' would be silly). Typically, *frequent* is a relative judgement. If we have a frequency list of phonemes from a phonemically transcribed corpus, we could say that phoneme X is one of the most frequent phonemes, based on its *relative* position on the list. We might also consider the frequency of phoneme X in this corpus *relative* to its frequency in some other corpus. Straightaway, we need an understanding of how to calculate relative (or *normalised*) frequencies, and we may well also need an understanding of how to

apply a statistical significance test to differences in frequency, to assure ourselves that the results our statistics produce have not occurred by simple coincidence. We will return to both these procedures later on.

Corpus linguistics is not unique in linguistics in appealing frequently to statistical notions and tests. Psycholinguistic experiments, grammatical elicitation tests and survey-based investigations, for example, all commonly involve statistical tests of some sort being carried out. However, frequency data is so regularly produced in corpus analysis that it is rare indeed to see a study in corpus linguistics which does not undertake some form of statistical analysis, even if that analysis is relatively basic and descriptive, for example using percentages to describe the data in some way. To put it another way, any empirically based approach to linguistics which deals with large collections of data points may have cause to employ statistical analysis. Empiricism lies at the core of corpus linguistics, so its frequent recourse to statistical analysis is not surprising.

In this section we simply do not have the space available to provide readers with a working knowledge of the statistics that corpus linguists use, though readers interested in exploring this area should see Oakes (1998), Baayen (2008) or Gries (2009b). Rather, we will highlight some general techniques used in corpus linguistics and, where relevant, certain known problems with the applications of particular statistics in corpus linguistics.

2.6.1 Descriptive statistics

We have noted that most studies in corpus linguistics use basic *descriptive statistics* if nothing else. Descriptive statistics are statistics which do not seek to test for significance. Rather, they simply describe the data in some way. The most basic statistical measure is a *frequency count*, a simple tallying of the number of instances of something that occur in a corpus – for example, there are 1,103 examples of the word *Lancaster* in the written section of the BNC. We may express this as a *percentage* of the whole corpus; the BNC's written section contains 87,903,571 words of running text, meaning that the word *Lancaster* represents 0.013 per cent of the total data in the written section of the corpus. The percentage is just another way of looking at the count 1,103 in context, to try to make sense of it relative to the totality of the written corpus. Sometimes, as is the case here, the percentage may not convey meaningfully the frequency of use of the word, so we might instead produce a *normalised frequency* (or relative frequency), which answers the question 'how often might we assume we will see the word per x words of running text?' Normalised frequencies (*nf*) are calculated as follows, using a *base of normalisation*:

$$nf = (\text{number of examples of the word in the whole corpus} \div \text{size of corpus}) \times (\text{base of normalisation})$$

So, if we want to see how often *Lancaster* would be expected to occur, on average, in each million words of the BNC, we set the base of normalisation to 1,000,000 and the calculation would be as follows:

$$nf = (1{,}103 \div 87{,}903{,}571) \times 1{,}000{,}000$$

In this case, $nf = 12.55$. Normalised frequencies based on 'occurrences per thousand words' or, as here, 'occurrences per million words' are the most commonly encountered in the literature; many corpus search tools generate these figures automatically. Note that, in fact, a percentage is simply a type of normalised frequency, where the base of normalisation is 100.

Corpus linguists often compare two or more corpora. Obviously generating normalised frequencies for the corpora being compared is essential when doing this. If we look up *Lancaster* in the BE06 corpus (Baker 2009), we find *Lancaster* occurs only ten times. BE06 is only 1,146,597 words in size, so the raw frequency count does not tell us if the word is more or less common in BE06 compared to the 1,103 instances in the BNC. A normalised frequency does tell us this – the word occurs 12.55 times per million words in the BNC but 8.72 times per million words in BE06. We can calculate a *ratio* to indicate how many times more often the word occurs in the BNC. This is done by dividing the larger number by the smaller – this results in the ratio 1.44, allowing us to say that for every occurrence of the word *Lancaster* in BE06 corpus, the word occurs 1.44 times in the written section of the BNC.

A special type of ratio called the *type–token ratio* is one of the basic corpus statistics. A *token* is any instance of a particular wordform in a text; comparing the number of tokens in the text to the number of types of tokens – where each *type* is a particular, unique wordform – can tell us how large a range of vocabulary is used in the text. We determine the type–token ratio by dividing the number of types in a corpus by the number of tokens. The result is sometimes multiplied by 100 to express the type–token ratio as a percentage. This allows us to measure vocabulary variation between corpora – the closer the result is to 1 (or 100 if it's a percentage), the greater the vocabulary variation; the further the result is from 100, the less the vocabulary variation. Since the size of the corpus affects its type–token ratio, only similar-sized corpora can be compared in this way. For corpora that differ in size, a normalising version of the procedure (*standardised type–token ratio*) is used instead.

You may have noticed something troubling about the statistics we have presented so far. While without doubt they are useful, interpreting them literally leads to absurdity. For example, in the case of the standardised frequency for the word *Lancaster*, it would be foolish to imagine that, if we chopped the BNC into 1-million-word chunks, we would with complete regularity find 12.55 occurrences of the word in each chunk. We clearly would not. The words in a corpus are not, in general, smoothly distributed through it, occurring at regular and precise intervals. In the case of the word *Lancaster*, we would probably expect to find lots of instances bunched together in a small number of texts where the city of

Lancaster is an important topic, and very few instances in the rest of the corpus. Normalised frequencies abstract from, and simplify, the reality of 'what's there' in the corpus. For this reason, it is usually considered good practice to report *both* raw and normalised frequencies when writing up quantitative results from a corpus.

2.6.2 Beyond descriptive statistics

To better understand and characterise the frequency data arising from a corpus, corpus linguists appeal to statistical measures which allow them to shift from simply describing what they see to testing the *significance* of any differences observed. Most things that we want to measure are subject to a certain amount of 'random' fluctuation. We can use *significance tests* to assess how likely it is that a particular result is a coincidence, due simply to chance. Typically, if there is a 95 per cent chance that our result is *not* a coincidence, then we say that the result is significant. A result which is not significant cannot be relied on, although it may be useful as an indication of where to start doing further research (maybe with a bigger sample of data).

The two most common uses of significance tests in corpus linguistics are calculating keywords (or key tags) and calculating collocations. To extract keywords, we need to test for significance every word that occurs in a corpus, comparing its frequency with that of the same word in a reference corpus. When looking for a word's collocations, we test the significance of the co-occurrence frequency of that word and everything that appears near it once or more in the corpus. Both procedures typically involve, then, many thousands of significance tests being carried out. This is all done behind the scenes in those tools that support keyword and collocation extraction. When we wish to apply significance tests to *other* quantitative data extracted from a corpus, however, we cannot normally count on the analysis software to handle the details for us; we must carry out the procedure ourselves.

Different significance tests can be used, depending on what type of data we have, for instance the *chi-square* test, the *t-test* and the *log-likelihood* test (Dunning 1993). The results of these tests allow researchers to assert with a degree of confidence that the results of their analysis either are or are not significant. This provides a distinct advantage to the corpus linguist – statistics, rather than simply describing the data in the corpus, can begin the process of sorting out significant differences from non-significant differences, thereby focusing the work of the linguist in accounting for and explaining the data in front of them. However, the use of significance tests is, predictably, not quite as straightforward as this. Some statistical tests make certain assumptions about the data. For example, the chi-square test presupposes a so-called *normal distribution* of the data. Data has a normal distribution if most of the values cluster relatively tightly around a mean (average) value – a pattern which, when plotted on a graph, gives us the classic 'bell-shaped' curve. This is not true for language data. For example,

word frequencies do not follow a normal distribution. Rather, they typically produce a markedly positively skewed distribution, with a pronounced 'hump' of a few very high-frequency words, followed by a very long tail of lots and lots of low-frequency words. The log-likelihood test (Dunning 1993) is now preferred by some corpus linguists as it makes no assumption of a normal distribution. This is only one example of the known issues with the statistical tests available. Another is that the number of observed examples you have is crucial in some tests, with chi-square becoming unreliable when the number of examples is very small. Again, there are proposed solutions to such problems – including using different statistical tests, notably Fisher's Exact Test (see McEnery *et al*. 2006).

Another complication is that when we do multiple significance tests, we expect some of them to give a false result, just by chance. If we use the standard 95 per cent cut-off point for significance, we expect such a chance result on average one time in twenty. This is potentially a major issue for corpus linguistics since (as noted above) the most common applications for significance testing, namely keywords and collocations, involve thousands of tests at a time. Typically, this problem is addressed by raising the bar for deeming a finding significant. Though 95% is standard in statistics, when keywords especially are calculated, much more stringent cut-off points are used, such as 99.9% or 99.99%. In fact, the default cut-off point for keywords in WordSmith is 99.9999%! Even at such a high level of significance, in most cases plenty of words are still found to be significant keywords.

A general principle emerges from these observations. Much as with descriptive statistics, the *inferential* statistics that allow us to test significance should be used with caution. Taking advice from a statistician, or at the very least consulting books such as Oakes (1998) or Woods *et al*. (1986) to gain an understanding of the limitations, if not the intimate workings, of the test you wish to use, is good practice for any researcher.

Taken together, significance tests form one of the groups of advanced statistics used most often in corpus linguistics. The other main group consists of techniques for *exploring* quantitative data, and investigating structure and relationships within it, rather than testing the significance of a particular result. Many such exploratory procedures exist. Perhaps the most important vis-à-vis corpus linguistics are *factor analysis* and *cluster analysis*. Factor analysis can be used when a large number of different quantitative measures have been made on a particular corpus (for instance, the frequencies of many different grammatical features). The purpose of a factor analysis is to determine which of a large number of quantitative variables are related to each other – a pair of variables are considered related if a change in one always means a change in the other. By determining relationships among variables, factor analysis reduces the number of variables that need to be taken into account (on the principle that two variables that are strongly related to one another are really only one factor of variation). Factor analysis was introduced to the mainstream of corpus linguistics by Biber (1988), who uses it as the basis of the multi-dimensional (MD) method. We will explore the MD approach in detail in Chapter 5. The other commonly used

exploratory technique is cluster analysis, which also investigates the structure of a large group of measurements. For example, imagine you have compiled several different statistics on the usage of a set of different verbs – how often they are used transitively; how often they are past tense; how often the subject is a human being; and so on. It would be quite tricky to tell, just by inspecting your tables of figures, which verbs are similar to one another. Two verbs having very close scores on one measurement doesn't mean they will necessarily be similar on other measurements. Cluster analysis groups the data – the verbs, in this case – according to similarity, taking *all* the information into account. Not only are clusters generated ('clusters' being groups of things, here verbs, that are statistically similar), but the statistical relationships between the clusters are also extracted, showing which are close together and which are relatively further apart. While there are many different mathematical techniques for clustering data, the general principle, of grouping the data according to similarities, is the same in each case. Cluster analysis has been employed for a number of different purposes in corpus linguistics; one notable recent approach is that of Divjak and Gries (2006), who use clustering to produce what they call *behavioural profiles* of words with multiple senses or sets of semantically similar words (see section 7.5.2). The example we gave above, of clustering verbs according to usage features, was effectively a toy example of Divjak and Gries' procedure.

This review of statistics in corpus linguistics is far from complete. However, it does at least serve to show the basic types of statistical analyses that are used in corpus linguistics – descriptive statistics, significance tests and exploratory techniques – and as such will provide a useful, if basic, guide to some of the studies discussed later in this book which draw upon these analyses. In particular, collocation as discussed in Chapter 6 draws heavily on significance tests, making the brief review of it presented above, at the very least, a necessary prerequisite to that discussion.

2.7 Summary

This chapter has reviewed the annotation and exploitation of corpus data. In doing so we have explored a number of objections to the use of corpus data in linguistics. Similarly, we have touched upon objections to the use of annotated corpus data. In both cases our defence is simple – the use of corpora and the use of annotations is not simply justifiable, it represents good scientific practice. As the telescope is indispensable to the astronomer, so the corpus is indispensable to the linguist. As keeping a careful record of analyses is important in any science, so it is in linguistics. However, as important as the use of the corpus in linguistics is, it is worth noting that other methods may also be gainfully combined with or used in place of the corpus – an idea to which we will return in Chapters 8 and 9. The key is to understand the strengths and weaknesses of a range of methods and to deploy them singly or in combination as appropriate. Finally, as has also been

discussed, the tools and statistical procedures that may be used with the corpus are important in determining how useful it can be. They support the exploration of specific research questions with a corpus, and in their absence the usefulness of the corpus itself is either severely constrained or negated.

In the next chapter we will explore two further areas which are important in the construction and use of corpora, namely legal and ethical issues. In discussing legal issues we will focus on building corpora from sources on the web. This will lead to a discussion of some of the advantages and disadvantages of the web as a source of corpus data.

Further reading

Surprisingly little has been written which focuses exclusively on the development of the tools of corpus linguistics. Some books have dealt with the basics of manipulating corpora, such as Barnbrook (1996) and McEnery *et al*. (2006). Other books have focused upon programming techniques that may be used to exploit corpora, notably Mason (2001) and more recently Weisser (2009) and Gries (2009a). However, there are no full-length works which deal comprehensively with markup conventions for corpus linguistics, perhaps understandably: the topic is rather dry and the techniques which are used are often generic techniques for which a literature exists already, as is the case with XML for example. But writing at a shorter length, Burnard (2005) and Wynne (2005) explain some of the benefits associated with the use of XML for corpus formatting, and, in the case of Burnard, provide some examples.

While it is perhaps unsurprising that not much has been written on markup conventions in corpus linguistics, it is less clear why relatively little has been produced looking at the process of corpus annotation. Many papers exist which describe individual corpus annotation schemes (e.g. Brugman *et al*. 2002; Hardie *et al*. 2009) or tagging systems (e.g. for part-of-speech tagging, see DeRose 1988; Brill 1995; Karlsson *et al*. 1995). But the only book which seeks to give the topic a relatively comprehensive treatment is Garside *et al*. (1997). Accordingly this book is strongly recommended as a reference point for the beginnings of a deeper exploration of corpus annotation. Van Halteren (1999) is less broad, addressing only one form of annotation (part-of-speech tagging), but is much more advanced and comprehensive within that purview.

The use of statistical techniques in corpus linguistics is increasingly common, and readers who wish to explore corpus linguistics further should certainly acquire a basic grasp of descriptive statistics and at least seek to understand the principles behind, if not the mechanics of applying, some of the more advanced statistics mentioned in this chapter. Oakes (1998) remains a good resource for those interested in statistics in corpus linguistics, though the book is best used as a reference resource and guide; while it serves as an excellent introduction to many statistics currently used in corpus linguistics, unless the reader is seeking

an extremely detailed understanding of the area, we would advise against a cover-to-cover reading. More recent books on the topic, notably the work of Baayen (2008) and Gries (2009b; see also Gries 2010a), should also be used in the same way by the great majority of readers.

Practical activities

(A2-1) In BNCweb/CQPweb, the query to find 'all words in the corpus ending in *-ness*' is

***ness**

Find out what search pattern you would need to use for the same query in another concordancer you have access to, such as Mark Davies' corpus.byu.edu interface, or WordSmith or AntConc.

(A2-2) Optional or alternative queries can be used in many ways – one use is to take into account the possibility of spelling variation, for example *colour* versus *color* in British and American English. In your concordancer, how would you create a search to retrieve all examples of both *colour* and *color*? (Note: in most concordancers, there is actually more than one way to do this.)

(A2-3) Searching for part-of-speech categories and other tags is often a bit more involved than searching for wordforms or phrases. Get a small amount of tagged data (or tag a raw text file of your own using a system such as the CLAWS web-tagger)[21] and use a third-generation concordancer to search for a word, such as *record* or *convert* in English, that can be a noun or a verb, depending on context. How can you control whether you find nouns, verbs, or both?

Note: depending on what concordancer you are using and what format your tagged data is in (e.g. XML versus *word_TAG* format), you may need to adjust the concordancer's options as well as your search pattern.

Questions for discussion

(Q2-1) Imagine you want to make use of a spoken corpus as part of a piece of research into pragmatics. One common way to go about this is to manually add some pragmatic annotation to the corpus as the first stage of analysis. What pragmatic features might you choose to annotate in a spoken corpus? What kind of classification scheme could you use to label these features? And how could you encode your labels into the corpus? (Hint for those unfamiliar with pragmatics: think about things like speech acts and politeness!)

(Q2-2) Part-of-speech tagging has, usually, around a 3% to 5% error rate. In what kinds of situation might this be a problem when doing research with a tagged corpus? How could you make allowance for these problems?

(Q2-3) Think of as many examples as you can of syntactic structures that have more than one possible analysis in English or another language you know well – either because they are ambiguous, or because different theories of grammar account for them in different ways. In a text with constituency parsing (i.e. phrase-structure brackets), what implications for the placement (and type) of phrase-boundaries would these differences of analysis have? Can you think of any cases where corpus searches or frequency counts might be altered by the decisions made about such structures?

(Two examples to get you started: (a) in *there-is*-type sentences in English, the *there* can be analysed as a subject noun phrase, a clausal adverbial or a unique 'existential' marker; each of these analyses could imply a different layout of parsing brackets! (b) subordinate clauses beginning in *where* can be analysed as adverbial clauses or relative clauses, depending on context, but different analysts draw the boundary in different places.)